**数据挖掘课程论文**

徽标

描述已自动生成

**Mechanisms of Action (MoA) Prediction**

|  |  |  |
| --- | --- | --- |
| **姓名** | **班级** | **学号** |
| **王子正** | **统计1803** | **8201180805** |
| **王泽瑄** | **统计1803** | **8201180828** |
| **田 庚** | **统计1801** | **8201180728** |

**Acknowledgement**

<https://www.kaggle.com/gunesevitan/mechanisms-of-action-moa-prediction-eda>

<https://www.kaggle.com/isaienkov/mechanisms-of-action-moa-prediction-eda>

<https://medium.com/swlh/drug-discovery-with-neural-networks-a6a68c76bb53>

<https://www.kaggle.com/kushal1506/moa-pytorch-feature-engineering-0-01846>

<https://www.ritchieng.com/machine-learning-dimensionality-reduction-feature-transform/>

<https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html>

<https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.html?highlight=bce#torch.nn.BCELoss>

<https://www.kaggle.com/thehemen/pytorch-transfer-learning-with-k-folds-by-drug-ids>

<https://www.kaggle.com/c/lish-moa/discussion/195195>

**摘要**



[https://www.kaggle.com/zizheng](https://www.kaggle.com/maomao99)

本文是kaggle竞赛Mechanisms of Action (MoA) Prediction比赛总结，这是我第一次参加kaggle竞赛，solo参赛取得了银牌成绩（180/4373）,参赛过程中参考了许多公开解法，已放在Acknowledgement部分。本文通过各种输入特征，如基因表达数据和细胞活力数据预测不同样本(sig\_id)的作用机制(MoA)反应的多个靶标概率，进行探索性数据分析，并采取了quantile transformer，one-hot encoding，PCA,K-Means、提取统计量特征等方法进行特征工程。建立了三个模型，其中模型一通过transfer learning的方法对pretrained的DNN模型进行微调；模型二为tabnet，本文也将深入讲解TabNet的网络结构和特点；模型三融合4个DNN模型和一个Resnet模型变体。最后融合这三个模型，在测试集得到的log loss为0.016138。

目录

[1 项目背景 4](#_Toc71363171)

[1.1 什么是MoA 4](#_Toc71363172)

[1.2 如何确定新药物的MoA 4](#_Toc71363173)

[2 数据介绍 4](#_Toc71363174)

[3 数据预处理/探索性数据分析 5](#_Toc71363175)

[3.1 分类特征 5](#_Toc71363176)

[3.2 数值特征 6](#_Toc71363177)

[3.3 目标变量 9](#_Toc71363178)

[3.4 目标变量与特征 13](#_Toc71363179)

[3.5 drug\_id 15](#_Toc71363180)

[4 特征工程 15](#_Toc71363181)

[4.1 统计特征 17](#_Toc71363182)

[4.2 特征处理总结 18](#_Toc71363183)

[5 评价标准及优化函数 18](#_Toc71363184)

[6 模型 19](#_Toc71363185)

[6.1 训练/验证集划分 19](#_Toc71363186)

[6.2 模型一 19](#_Toc71363187)

[6.3 模型二 20](#_Toc71363188)

[6.4 模型三 24](#_Toc71363189)

[6.5 模型融合 24](#_Toc71363190)

[7 总结 31](#_Toc71363191)

[参考文献 31](#_Toc71363192)

[附录 32](#_Toc71363193)

[数据集 32](#_Toc71363194)

[代码 32](#_Toc71363195)

## 1 项目背景

Connectivity Map是由麻省理工学院和哈佛大学的Broad研究所、哈佛大学创新科学实验室（LISH）和美国国立卫生研究院（NIH）等共同发起的一个项目。该项目举办了这次kaggle竞赛，其目标是通过改进药物成分作用机制（MoA）预测算法来推进药物开发。

### 1.1 什么是MoA

过去，科学家们从天然产品中提取药物，或者从传统疗法中获得灵感。非常常见的药物，如扑热息痛（paracetamol），在人们了解其药理作用的生物学机制几十年前就已经投入临床使用。今天，随着更强大技术的出现，药物发现已经从过去的偶然方法转变为基于对疾病潜在生物学机制的理解的更有针对性的模型。在这个新的框架中，科学家们试图识别与疾病相关的蛋白质靶标，并开发一种能够调节该蛋白质靶标的分子。作为对特定分子生物活动的速记，科学家们给分子贴上了一个标签，简称为作用机制(MoA)。药理学中，MOA指的是药物可以产生药理学效果的生化反应，一般都是绑定到一个分子目标上，例如酶或者是受体，也有不绑定受体直接和身体作用的，例如抗酸剂或泻药。MoA描述的是细胞级的功能或解剖的变化。 ​​​​

### 1.2 如何确定新药物的MoA

一种方法是用这种药物处理人类细胞样本，然后用算法分析细胞反应，算法可以在大型基因组数据库中搜索与已知模式的相似性，比如已知MoA药物的基因表达库或细胞存活模式库。

## 2 数据介绍

我们需要根据各种输入特征，如基因表达数据和细胞活力数据预测不同样本(sig\_id)的作用机制(MoA)反应的多个靶标概率。

* train\_features.csv:训练集对应的特征数据。特征g-表示基因表达数据，c-表示细胞活力数据。cp\_type表示使用化合物(cp\_vehicle)或控制扰动(ctrl\_vehicle)处理的样品;控制扰动无MoA;cp\_time和cp\_dose表示治疗时间(24、48、72小时)和剂量(高、低)。
* train\_drug.csv 训练集样本的sig\_id和drug\_id对应关系
* train\_targets\_scored.csv：MOA的二分类标签数据
* train\_targets\_nonscored.csv：额外没有带有标签的MOA数据，在测试数据集中不存在
* test\_features.csv：测试数据的特征。需要选手预测测试数据中每一行的每个MoA得分的概率
* sample\_submission.csv：提交文件

## 3 数据预处理/探索性数据分析

训练与测试特征集共占用内存186.2MB。其中训练数据集共23814条数据，876列；测试数据集共3982，876列。训练集与测试集使用的特征完全相同，有872个浮点型特性，1个整型特征（cp\_time）,3个类别型特征。

数据集没有缺失值，是很规整的数据集。简单探查后发现特征没有不合乎逻辑的异常值，数值特征中有一些偏差很大的数据，但由于对生物实验原理理解不够，暂时不作处理。下面将针对不同类型特征进行分析。

表格

描述已自动生成

### 3.1 分类特征

共有四个类别型特征，sig\_id、cp\_time、cp\_type和cp\_dose，sig\_id为数据标识不用分析。

绝大多数数据的cp\_type都为trt\_cp，只有8%的数据为ctl\_vehicle(可以理解为对照组)，且数据的相对分布在训练集和测试集上都是相同的。

图表, 条形图

描述已自动生成

Figure 1 cp\_type countplot

治疗时间(treatment duration)取值为24，48和72小时，分布十分均匀。

**图表, 条形图

描述已自动生成**

Figure 2 cp\_time countplot

药物剂量也十分均匀。

**图表, 条形图

描述已自动生成**

Figure 3 cp\_dose countplot

### 3.2 数值特征

数值型特征包括基因表达(gene expression，数据集中前缀为g-)和细胞活性(cell viability,数据集中前缀为c-)，其中基因表达特征共有772个，细胞活性特征共有100个。

3.2.1 细胞活性

在细胞群体中总有一些因各种原因而死亡的细胞，总细胞中活细胞所占的百分比叫做细胞活力。数据集中共有100个细胞活性特征（c-0, ……c-99），每个细胞活性特征都是在相同的细胞基准水平上测得的。

选择15个细胞活性特征查看分布，可以看出训练集和测试集的细胞活性集中在均值0附近，且在负值部分也有一个集中趋势，整体呈现左偏趋势。

图片包含 图表

描述已自动生成

图片包含 图表

描述已自动生成

图表

中度可信度描述已自动生成

Figure 4 c- distribution

查看不同特征间的相关情况，由热力图颜色深度可以看出特征间相关性很是很强的，在特征工程中是很有用的信息。

图表

描述已自动生成

Figure 5 c- correlation heatmap

深入分析，选取相关性最高（均大于0.9）的20对细胞活性特征，显然可以利用特征间的相关性构造新的特征，强化特征的预测能力(prediction power)。

表格

描述已自动生成

3.2.2 基因表达

基因表达（gene expression）是指将来自基因的遗传信息合成功能性基因产物的过程。数据集中共有772个基因表达特征(g-0,……g-771)。

随机选取15个基因表达特征，可以看出相比细胞活性特征分布的同质性，基因表达特征在分布上更有特点，峰度和偏度均有明显不同，在特征工程过程中应该尽量利用到这些特征。

图表

描述已自动生成

图表

描述已自动生成

图表

描述已自动生成

Figure 6 g- distribution

查看基因表达特征间的相关关系，可以看出绝大多数特征对的相关系数颜色很浅，均在0.25左右，相比细胞活性特征可利用性不强。

图表

中度可信度描述已自动生成

Figure 7 g- correlation heatmap

### 3.3 目标变量

目标变量被分为两组，non-scored和scored，每种药物在这些目标变量上去值为0或1，这些目标变量是科学家们对每种药物提供的批注。Non-scored变量只在训练数据中存在，比赛中是以得分变量的预测正确情况作为评价标准，但我们仍然可以利用非得分变量进行探索性分析，特征工程和建模。

得分目标共有206个，非得分特征共有402个，总共608个目标变量。

表格

描述已自动生成

手机屏幕截图

描述已自动生成

查看样本目标变量的取值情况，可以看出对非得分变量和得分变量，每个样本均可取0或多类，即这是一个multi-label的分类问题，同时样本也可不属于已有变量中的任何一类。大多数情况下，样本被分为1类或0类，但是在训练集上也有一小部分样本被分为2,3,4,5或7个不同的标签。但非得分和得分标签分布情况不同，可以看出得分标签被分为1类的要比0类多，但非得分标签被分为0类要比其他情况多，下面将针对得分和非得分标签分别分析。

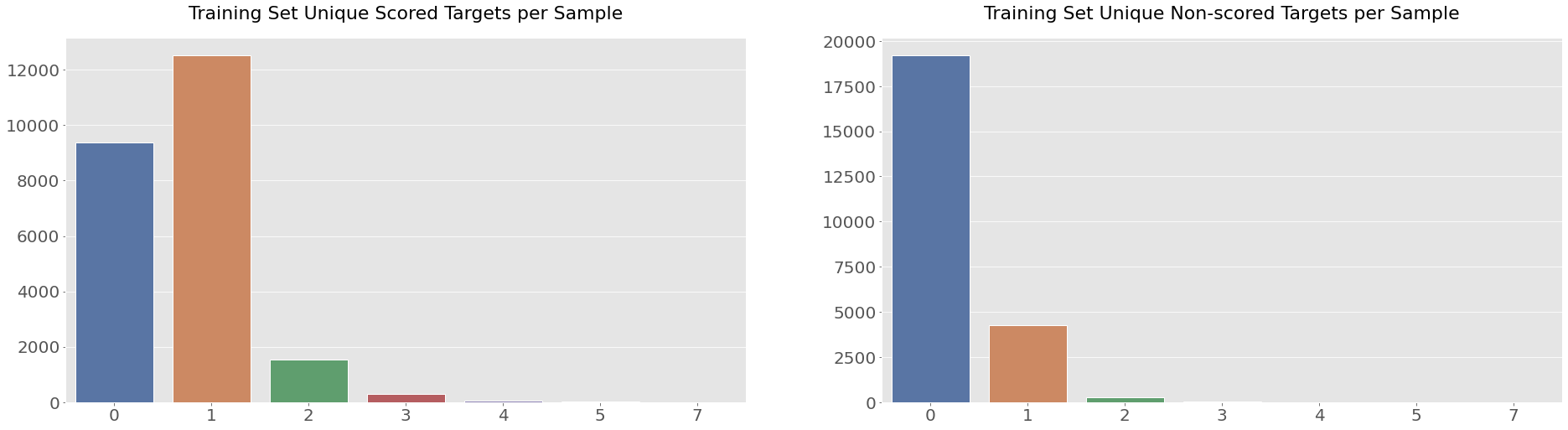


Figure 8 target样本分布情况

3.3.1 得分标签

可以看出，最普遍的得分标签为nfkb inhibitor、proteasome inhibitor, cyclooxygenase inhibitor, dopamine receptor antagonist, serotonin receptor antagonist 和dna\_inhibitor，分别有400多个样本被标为这些标签。最少见的标签为atp-sensitive potassium channel antagonist和erbb2 inhibitor，每个标签仅有一个样本被归为这些标签。可以预见测试集也会是类似的分布情况。同时还可以看出很多得分标签分类的次数是相同的，这些标签间可能有一定的关系。

3.3.2 非得分标签

最普遍的非得分标签为ace inhibitor, purinergic receptor antagonist, map kinase inhibitor, sterol demethylase inhibitor，分别有70多个样本被归为这些标签。同时有71个非得分标签没有在任何样本上取得。非得分标签的计数情况和得分标签相差很大，训练集上很多样本并没有归为任意一类。非得分标签计数相同的情况比得分标签更明显，标签间应该有更紧密的关系。

3.3.3 目标变量相关情况

大多数得分标签和非得分标签的相关系数都接近于0，但是从热力图上的红点可以看出还是有一些变量对的相关系数很大（热力图上的白线代表缺失值，并不是相关系数为1）。

电脑屏幕的照片

中度可信度描述已自动生成

Figure 9 得分变量计数

日程表

描述已自动生成

Figure 10 非得分变量计数

图形用户界面

描述已自动生成

Figure 11 目标变量相关系数热图

### 3.4 目标变量与特征

3.4.1 得分变量与数值型特征

查看每个得分标签与所有的数值特征的相关关系,并找出每个得分标签对应的绝对值最大的相关系数，可以看出有很多得分标签对任意一个数值型特征相关性都很弱，但有少数标签的最大相关系数超过了0.9。

随机选取部分细胞活性和得分标签绘制散点图，可以看出大多数情况下细胞活性与得分标签间存在正相关关系，但是部分特征与标签存在负相关或无明显关系。

同理绘制基因表达与得分标签散点图，相比细胞活性，基因表达与得分标签的相关性很弱，标签取值为1的数据点在水平方向上分布更均匀。所有的基因特征和细胞活性特征在标签取值为1时都聚集在0附近。基因表达特征取值很大(>2或<-2)表示药物对当前细胞影响显著，取值接近于0意味着药物对细胞没有显著影响，可以忽略不计。

同理可以对非得分标签以及分类特征的相关情况进行分析，可查看Acknowledgement，这里不再赘述。

表格

描述已自动生成

图表, 日程表

描述已自动生成

Figure 12 得分变量与特征相关情况

图形用户界面, 应用程序

描述已自动生成

图形用户界面, 应用程序

中度可信度描述已自动生成

图片包含 应用程序

描述已自动生成

Figure 13 c-特征与得分变量

图形用户界面

低可信度描述已自动生成

图表

低可信度描述已自动生成

图形用户界面, 应用程序

描述已自动生成

Figure 14 g-特征与得分变量

### 3.5 drug\_id

举办方在比赛临近结束时加入了新的drug\_id数据，只在训练数据集中存在，通过查看drug\_id的技术情况，可以看出绝大多数drug\_id在数据集中出现了18次及以下，且集中在6次左右。

图表, 条形图, 直方图

描述已自动生成

Figure 15 drug\_id分布情况

## 4 特征工程

由于生物实验数据获得原理的特殊性，绝大多数实验数据都使用quantile normalization将特征缩放在同样的分布或范围下。Quantile normalization通过执行一个秩转换能够使异常的分布平滑化，并且能够比缩放更少地受到离群值的影响。但是，它会使特征间及特征内的关联和距离失真。QuantileTransformer函数提供了一个基于分位数函数的无参数转换，将数据映射到了零到一的均匀分布或高斯分布上。这种scaling方法在kaggle竞赛上十分常见，并且通过实验证明了对特定的数据, quantile normalization效果比其他scaling方法更好，本文中即对基因表达特征和细胞活性都使用了该方法，转换后的分布如图,可以看出数据分布为高斯形状，但仍然可以看出基本的统计特征。算法原理可见：

<https://academic.oup.com/bioinformatics/article/19/2/185/372664>

图标

描述已自动生成

Figure 16 标准化后c-特征分布

主成分方法是一种常用的降维方法，每一个主成分都寻求在最大程度代表原变量，并与其他主成分无关。但在本文采取主成分方法分别对基因表达和细胞活性特征提取主成分，并将这些主成分通过聚类方法分类标签，以标签作为新的分类特征（这一步应该是有data leakage的，但由于测试集和训练集在特征上分布情况大致相同，影响不是很大；且该方法如果在训练和测试集上分别进行，主成分和聚类标签的意义在两个集合上不同，作为新的特征也没有可解释性）。

通过尝试，对基因表达特征选取600个主成分，累计方差贡献率0.9486；对细胞活性特征选取50个主成分，累计方差贡献率0.8166，经过PCA的数据情况如下：

图片包含 文本

描述已自动生成

对提取的主成分进行K-Means方法聚类，采取Elbow Method确定聚类，数，可以看出，肘点为5，即将主成分聚为5类。再用PCA方法降聚类后的主成分降为二维平面可视化。

图表, 折线图

描述已自动生成图表, 气泡图

描述已自动生成

Figure 17 聚类肘部法则与降维后可视化

同理也可以直接用原特征进行聚类，这里根据方差阈值过滤掉方差在0.85以下的特征，对0.85以上的特征进行聚类，这些高方差的特征更具有prediction power，聚类后的效果也更明显。同上根据肘部法则分别对基因表达特征和细胞活性特征进行K-Means聚类，对基因表达特征采取聚类数为22，细胞活性特征聚类数为4。

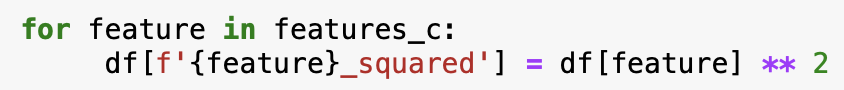
### 4.1 统计特征

①由3.2.1，将相关系数最强的20对细胞活性特征相乘获得组合特征，比如c-52\*c-42=c52\_c42。

图片包含 表格

描述已自动生成

②将所有细胞活性特征乘方，得到新特征。



③将kaggle讨论区获得的Resnet模型中预测能力很强的部分基因表达特征乘方得到新特征。

文本

描述已自动生成

④以每个样本为粒度，分别获得基因表达特征、细胞活性特征以及两特征总体的统计量，包括总和、标准差、偏度和峰度。

文本

描述已自动生成

### 4.2 特征处理总结

完成以上特征处理后将cp\_type为控制组取值ctl\_vehicle的样本去掉，并删除cp\_type列。

|  |  |  |  |
| --- | --- | --- | --- |
| 原特征 | 特征处理(按顺序) | 新特征 | 特征数 |
| c-0 to c-99 | QuantileTransformer | c-0 to c-99 | 100 |
| g-0 to g-771 | QuantileTransformer | g-0 to g-771 | 772 |
| c-,g- | PCA+K-Means | clusters\_pca\_0 to clusters\_pca\_4 | 5 |
| c-(Variance 0.85+) | K-Means | clusters\_c\_0 to clusters\_c\_3 | 4 |
| g-(Variance 0.85+) | K-Means | clusers\_g\_0 to clusters\_g\_21 | 22 |
| c- | Square | c\_squared | 100 |
| g-574,g-211…… | Square | g\_squared | 45 |
| c-52&c-42…… | Multiply | c52\_c42…… | 20 |
| c-,g-,c-&g- | Sum,mean,std,kurt,skew | NA | 15 |
| cp\_type | Drop ctl\_vehicle |  |  |
| cp\_time,cp\_dose | One-hot encoding |  |  |

表格 1 特征工程

## 5 评价标准及优化函数

从EDA过程中可以看出，这是一个Multi-label多标签分类问题，且每个样本可以被分为0类或多类标签，举办方确定的评价标准为log loss。对每一行，需要预测每个样本被分为某一标签的概率。若有N行M个得分标签，就有N\*M个预测。

* N为行数
* M为目标变量数(m=1,2,…M)
* 为第i行第m列的预测概率
* 为第i行第m列的真实值（1或0）
* 为自然对数

该损失函数在pytorch中可以使用BCEWITHLOGITSLOSS函数直接实现。

但在训练集上如果直接使用log loss作为目标函数进行训练，有可能会出现过拟合的情况，尤其是在本问题中数据量相对较少，模型很容易对预测结果过于自信，导致在测试集上的表现变差，因此考虑对得分目标进行label smoothing（label smoothing是对真实值操作的，与预测结果无关）。针对multi-label的二分类问题，平滑公式如下：

为平滑指数，取值范围为[0,1]，可以看出当时即为原标签，时恒为0.5。可以看出相比未平滑的标签只能取值为0或1的one-hot encoding，label smoothing缩小预测值与真实值之间的“差距”，防止模型在训练过程中过于自信，导致通用能力（在测试集上预测）变差。

## 6 模型

本文使用了三个模型，将每个模型的预测结果加权平均得到最终预测。

### 6.1 训练/验证集划分

这是一个多标签的分类问题，为确保划分集合时每个标签在集合中的分布情况相同，需采用分层方法得到验证集，同时多标签问题需采用iterative stratification算法（算法原理可见下文P149，在python中有函数实现）。

<https://link.springer.com/content/pdf/10.1007%2F978-3-642-23808-6_10.pdf>

同时在竞赛过程中讨论区出现了根据drug\_id在训练集的出现频数再分层的方法，分别对drug\_id出现次数大于18次和小于等于18次进行分层，确保每一层出现次数多的drug\_id和出现次数少的drug\_id分布情况都相同。以此分为7折，每折都均匀地有7层。训练7个随机种子，每个随机种子训练7折，对每个种子每折得到的预测结果进行简单评价，得到模型的最终预测结果。

<https://www.kaggle.com/c/lish-moa/discussion/195195>

### 6.2 模型一

模型一Architecture如下，采取Adam优化器，超参设置可参考代码。在训练过程中使用了transfer learning的gradual unfreezing技术，即先对得分和非得分目标变量预先训练好一个表现最好的模型，然后只针对得分变量，从最后一层开始逐渐解冻模型，每次epoch内微调所有已解冻的层；然后解冻下一个较低的层，再微调；以上重复；直到微调所有的层，直到在最后一次迭代中收敛。最终得到模型一的cv log loss为0.015621。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | BN | Dropout | Activation | Size(in,out) |
| input |  |  |  |  |
| Layer 1 |  |  | Leaky ReLU | #feature,1500 |
| Layer 2 |  | 0.5 | Leaky ReLU | 1500,1250 |
| Layer 3 |  | 0.35 | Leaky ReLU | 1250,1000 |
| Layer 4 |  | 0.3 | Leaky ReLU | 1000,750 |
| Layer 5 |  | 0.25 | Weight\_norm | 750,#target |
| output |  |  |  |  |

表格 2 模型一结构

### 6.3 模型二

模型二使用了TabNet（<https://arxiv.org/pdf/1908.07442.pdf>）这篇论文提出的TabNet是一种针对于表格数据的神经网络，它通过类似于加性模型的顺序注意力机制（sequential attention mechanism）实现了instance-wise的特征选择，还通过encoder-decoder框架实现了自监督学习，从而将树模型的可解释性与DNN的表征能力很好地结合到了一起，相信这种兼具两者优点的模型将会成为数据挖掘竞赛中的一大利器，也对未来的研究提供了一个很好的思路。模型的结构如下图，encoder部分可以由多个step顺序构成（当然step越多越容易overfitting）。feature transformer模块用于特征计算，包含两个部分，一部分仅在一个step实现，另一部分对每个step共享，同时加入skip connection加快训练。Attentive transformer模块则用于计算当前step的Mask层（每个Mask层都包括上一个step得到的信息），可以理解为每个step特征的局部重要性，通过Mask层与当前step输出进行乘法并对每个step加和即可得到feature attributes即特征的全局重要性。每个step的输出加和后经过FC层即可得到encoder部分的最终输出。具体的算法原理可参考论文，项目中使用了pytorch封装的tabnet模块，超参可见代码（只使用了一个step），得到CV log loss=0.015814。

图示

描述已自动生成

Figure 18 TabNet网络结构

图形用户界面, 图示

描述已自动生成

Figure 19 Feature transformer和Attentive transformer结构

下面将深入探讨Tabnet的网络结构。

* N层：即batch normalization层
* Feature transformer层：其作用与之前的FC层类似，都是做特征计算，只不过复杂一些。

其中GLU是gated linear unit，它其实就是在原始FC层的基础上再加上一个门控，其计算公式为  。

可以看出Feature transformer层由两个部分组成，前半部分层的参数是共享的，也就是说它们是在所有step上共同训练的；而后半部分则没有共享，在每一个step上是分开训练的。这样做是考虑到对于每一个step，输入的是同样的features（Mask层只是屏蔽了一些feature，并没有改变其它feature），因此我们可以先用同样的层来做特征计算的共性部分，之后再通过不同的层做每一个step的特性部分。另外，可以看到层中用到了残差连接，乘  是为了保证网络的稳定性。

* Split层：该层比较简单，就是将Feature transformer层输出的向量切成两部分，用公式表示为  ，其中 用于计算模型的最终输出，而 用来计算下一个step的Mask层。
* Attentive transformer层：该层的作用是根据上一个step的结果，计算出当前step的Mask层，具体结果如Figure 19 (d)

其中Sparsemax层可以理解为Softmax的稀疏化版本，可以参看Sparsemax原论文[6]。不像Softmax的平滑变换，Sparsemax通过直接将向量投影到一个simplex来实现稀疏化，其计算公式为：

根据Attentive transformer层的结构，可以将其计算公式写做：

其中是上一个step中Split层划分出来的,代表FC+BN层，而是Prior scales项，其形式为

它用来表示某一个feature在之前的step中的运用程度，按照正常的直觉，如果一个feature已经在之前的step中用了很多次，那么它就不应该再被模型选中了，因此模型通过这个Prior scales项来减小这类feature的权重占比，从式子可以看出，如果令  ，那么每个feature只能被用一次，而当增加时，这个约束就会变“软”一些。最后因为得到的Mask矩阵是𝐵 维的，根据Sparsemax的性质有 ，因此可以理解为模型在当前step上，对于batch样本的注意力权重分配，值得注意的是，对于不同的样本，Attentive transformer层输出的注意力权重也不同，这个特点在论文中被叫做instance-wise。

接着是正则项，为了增强模型对feature稀疏选择的能力，这里额外引入一个正则项，具体形式如下：

这个正则项很好理解，就是算了一个平均的熵，其目的是希望的分布尽量趋近于0或1，而考虑到 ，则 反映的就是的稀疏程度，越小，越稀疏。

最后再看一下Feature attribute输出，它其实刻画的是feature的全局重要性。模型先对模型一个step的输出向量求和，得到一个标量，这个标量反映的是这个step对于最终结果的重要性，那么它乘以这个step的Mask矩阵就反映了这个step中每个feature的重要性，将所有step的结果加起来，就得到了feature的全局重要性。

总结一下，TabNet采用了顺序多步（sequential multi-step）框架，构造了一种类似于加性模型的神经网络，模型中比较关键的是Attentive transformer层和Feature transformer层，它们分别实现了下面两种功能：

* 特征选择：Attentive transformer层可以根据上一个step的结果得到当前step的Mask矩阵，并尽量使得Mask矩阵是稀疏且不重复的。值得注意的一点是，不同样本的Mask向量可以不同，也就是说TabNet可以让不同的样本选择不同的特征（instance-wise），而这个特点是树模型所不具备的，对于XGBoost这类加性模型，一个step就是一棵树，而这棵决策树用到的特征是在所有样本上挑选出来的（例如通过计算信息增益），它没有办法做到instance-wise。
* 特征计算：Feature transformer层实现了对于当前step步所选取特征的计算处理。还是类比于决策树，对于给定的一些特征，一棵决策树构造的是单个特征的大小关系的组合，也就是上面提到的决策流形，而之前那个简单神经网络就是通过一个FC层来模仿这个决策流形，但FC层只是构造了一组简单的线性关系，并没有考虑更加复杂的情况，因此TabNet通过更复杂的Feature transformer层来进行特征计算，个人感觉它的决策流形不一定和决策树的相似，在一些特征组合上它可能比决策树做得更好。

TabNet还有自监督学习方面的创新，训练时包括非监督的与训练和自监督的微调，如下图所示：

表格

描述已自动生成

Figure 20 自监督举例

简单来说，我们认为同一样本的不同特征之间是有关联的，因此自监督学习就是先人为mask掉一些feature，然后通过encoder-decoder模型来对mask掉的feature进行预测。我们认为通过这样的方式训练出来的encoder模型，可以有效地将表征样本的feature（可以理解为对数据进行了编码或压缩），这时再将encoder模型由于回归或分类任务，就能够事半功倍。自监督学习时的encoder模型就是上图中的模型。

下面将分析Figure 18（b）中的decoder结构。图中的encoded representation就是encoder中没有经过FC层的加和向量，将它作为decoder的输入，decoder同样利用了Feature transformer层，只不过这次的目的是将representation向量重构为feature，然后类似地经过若干个step的加和，得到最后的重构feature。

设在一开始对feature做mask的矩阵是，特征数据是 ，则encoder的输入是 ，若最后decoder的输出是 ，那么自监督学习就是减小真实值与重构值之间的差别，考虑到不同的feature的量级不一定相同，因此采用正则化后的MSE作为loss，形式如下：

另外，为了让模型学到的是整个feature数据的表征方法，而不仅仅是某些feature，在自监督学习的训练过程中，每一轮都会对矩阵重新采样，以此来保证encoder模型的整体表征能力。

作为总结，自监督学习可以让encoder模型学到feature数据的representation，通过representation再对任务进行分类和回归就会变得更加容易了。类似于NLP中的pre-training和fine-tuning过程，自监督学习可以很好地利用到没有label的数据，从而使得模型在有label的样本比较少时也能取得不错的效果，并且还能加快模型的收敛速度。

### 6.4 模型三

模型三融合了5个模型，一个为Resnet的变体模型（Resnet的input分为两路，一路为全部feature，另一路为fork来的powerful predictor），其它为简单的5层、4层、3层、2层DNN模型。下面详细给出Resnet变体和5层DNN的architecture，其余模型结构和5层DNN类似。最后的CV log loss为0.015693。

### 6.5 模型融合

以0.32，0.34，0.34的权重对三个模型预测概率加权进行模型融合（model blending），最终得到Private Leaderboard log loss为0.016138。当然也可以采用stacking，majority vote等集成方式，这里未作考虑。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | BN | Dropout | Act | Size(in,out) |
| input1 |  |  |  |  |
| Layer 1 |  | 0.2 | eLU | #feature,512 |
| Layer 2 |  |  | eLU | 512,256 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | BN | Dropout | Act | Size(in,out) |
| input2 |  |  |  | #Ppredictor |

Concatenate (256,#Ppredictor), 703)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | BN | Dropout | Act | Size(in,out) |
| Layer 3 |  | 0.3 | reLU | 703,512 |
| Layer 4 |  |  | eLU | 512,256 |
| Layer 5 |  |  | reLU | 256,256 |

Average (256,256), 256)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | BN | Dropout | Act | Size(in,out) |
| Layer 6 |  |  | seLU | 256,256 |
| Layer 7 |  |  | seLU | 256,206 |
| Layer 8 |  |  | sigmoid | 256,#target |
| output |  |  |  |  |

表格 3 Resnet结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | BN | Dropout | Act | Size(in,out) |
| input1 |  |  |  |  |
| Layer 1 |  | 0.4 | reLU | #feature,2560 |
| Layer 2 |  | 0.4 | reLU | 2560,2048 |
| Layer 3 |  | 0.4 | reLU | 2048,1524 |
| Layer 4 |  | 0.4 | reLU | 1524,1012 |
| Layer 5 |  | 0.4 | reLU | 1012,780 |
| output |  | 0.2 | reLU | 780,#target |

表格 4 5 layer DNN结构

图片包含 表格

描述已自动生成

Figure 21 5层DNN结构

表格

描述已自动生成

Figure 22 4层DNN结构

表格

描述已自动生成

Figure 23 3层DNN结构

表格

描述已自动生成

Figure 24 2层DNN结构

图示

中度可信度描述已自动生成

Figure 25 ResNet网络结构

## 7 总结

虽然第一次参赛就取得了银牌（180/4373），但这次比赛学习到的新知识显然要比一枚银牌更宝贵；特征工程不仅仅是简单地处理缺失值，one hot encoding甚至frequency encoding，我们可以通过对数据的探索性分析提取更为有效的信息，这次比赛也锻炼我的代码能力，通过写类将代码规范化，也练习了pytorch和tensorflow的常用功能；接触了全新的tabnet模型，在对表格数据建模时不再只能单一地使用xgboost，lightgbm，catboost等树模型的集成方法，可以将神经网络的各种架构融入到表格数据的建模中。

整理了一些排名靠前的解法，下面分模块总结winning solution的创新点。

①特征工程

* Log scaler；先按样本归一化再使用quantiletransformer；
* t-SNE降维
* 先对测试集通过1层的NN预测非得分变量，然后将非得分变量作为特征建模
* 去掉在实验组和对照组上差异较小的特征
* 用原始数据做data augmentation

②模型创新

* 模型融合使用算法估计权数分配
* 使用DeepInsight模型将结构数据转化为图像数据
* 使用CNN作为主要模型
* 注意力机制的DNN模型，在训练过程中随epoch增加batch size

## 参考文献

[1] Lamb J, et al. The Connectivity Map: using gene-expression signatures to connect small molecules, genes, and disease. *Science*. 2006/9/29. 313(5795):1929-35, (2006).

[2] B.M. Bolstad, R.A Irizarry, M. Åstrand, T.P. Speed, A comparison of normalization methods for high density oligonucleotide array data based on variance and bias, Bioinformatics, Volume 19, Issue 2, 22 January 2003, Pages 185–193, <https://doi.org/10.1093/bioinformatics/19.2.185>

[3] Sechidis K., Tsoumakas G., Vlahavas I. (2011) On the Stratification of Multi-label Data. In: Gunopulos D., Hofmann T., Malerba D., Vazirgiannis M. (eds) Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2011. Lecture Notes in Computer Science, vol 6913. Springer, Berlin, Heidelberg. <https://doi.org/10.1007/978-3-642-23808-6_10>

[4] [arXiv:1908.07442](https://arxiv.org/abs/1908.07442) [cs.LG]

[5] [arXiv:1906.02629](https://arxiv.org/abs/1906.02629) [cs.LG]

[6] arXiv:1602.02068 [cs.CL]

## 附录

### 数据集

<https://www.kaggle.com/c/lish-moa/data>

### 代码

<https://github.com/trinityWang/DA-DS-ML-DL/tree/main/kaggle%20moa>

iterative stratification伪代码

图片包含 日程表

描述已自动生成